Análise e Projeto Orientado a Objetos utilizando a Unified Modeling Language (UML)

Sumário

Histórico Linguagens OO

- Simula 67 (1964-1967)
- Puras
  - Smalltalk (1972)
    - ANSI Smalltalk (1998)
  - Eiffel (1986)
    - ECMA Eifffel (2005)
  - Self (1987)
    - Self 4.5.0 (2014)

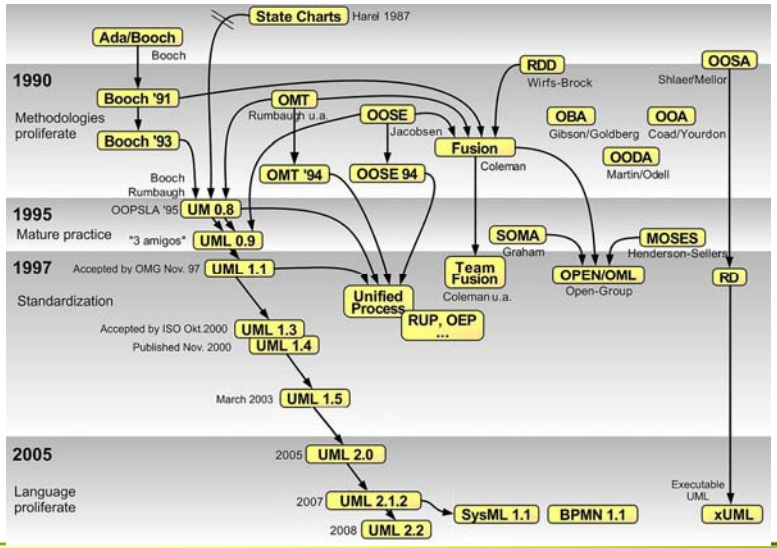
Histórico Linguagens OO

- Com conceitos/elementos OO
  - C++ (1983)
    - ISO/IEC C++ (2011)
  - C# (2000)
    - C# 6.0 (2015
  - Java (1995)
    - Java 8 (2015 Update 60)
  - JavaScript (1995)
    - ECMAScript 6 (2015)
  - Objective-C (1983)
    - Objective-C .21 (2009)
  - Python (1991)
    - Python 3.4.3 (2015)
  - Swift (2010)
    - Swift 1.2 (2015)
  - Scala (2003)
    - Scala 2.11.7 (2015)
  - Ruby (1993)
    - Ruby 2.2.2 (2015)

Histórico OO Métodos

- Shlaer-Mellor
  - Sally Shlaer e Stephen Mellor
  - 1988
  - Object-Oriented Systems Analysis (OOSA)
- OMT (Object Modeling Technique)
  - Rumbaugh, Blaha, Premerlani, Eddy and Lorensen
  - 1991
- OOSE (Object-Oriented Software Engineering)
  - Ivar Jacobson
  - 1992
- Booch
  - Object-oriented analysis and design (OOAD)
  - 1991
  - Grady Booch
- Coad-Yourdon
  - Peter Coad e Edward Yourdon
  - Object-Oriented Analysis (OOA)

Métodos de Análise e Projeto Orientado a Objetos - Evolução



Por que a Orientação a Objetos?

- As abstrações podem corresponder às “coisas” do domínio do problema, facilitando o entendimento
- Esta abstração facilita a comunicação com os usuários
- Os mesmos objetos existem em todas as fases e uma notação única facilita a INTEGRAÇÃO ENTRE FASES de desenvolvimento
- A tecnologia de objetos facilita o entendimento do domínio do problema, permitindo o GERENCIAMENTO DA COMPLEXIDADE através da modularização
- Facilidade de mudanças através do ENCAPSULAMENTO de dados

Por que a Orientação a Objetos?

- Capacidade de aproveitar novas plataformas e ferramentas
- Facilidade de manutenção
- Economia de custos
- Encapsulamento das aplicações existentes
- Melhores interfaces
- Maior produtividade
- Participação no "futuro da computação“
- Prova da capacidade de usar a tecnologia
- Rápido desenvolvimento de aplicações estratégicas
- Reuso de software

Por que a Orientação a Objetos?

- Domínio do Problema (Mundo Real)

- Domínio da Solução (Software)

Orientação a Objetos - Conceitos

- OBJETO
- MÉTODO
- MENSAGEM
- CLASSE
- CLASSIFICAÇÃO
- GENERALIZAÇÃO
- ESPECIALIZAÇÃO
- HERANÇA
- POLIMORFISMO
- SOBRECARGA
- ENCAPSULAMENTO
- ABSTRAÇÃO
- MODULARIZAÇÃO

OBJETO

- Entidades que possuem **dados e instruções** sobre como manipular estes dados
- Os objetos estão ligado à solução do problema.
  - Software Gráfico – Objetos: Círculos; Linhas; etc.
  - Software BD – Objetos: Tabelas; Linhas; Campos; etc.
  - Software Comercial: Pedidos; Produtos; Clientes; etc.
- Na OO a solução do problema consiste em um primeiro momento estabelecer quais os objetos serão necessários.
- Um objeto representa uma entidade: física, conceitual ou de software

OBJETO

- Os dados mantidos pelo objeto são chamados de atributos(propriedades)
- Os atributos de um objeto representam seu ESTADO, ou seja, o valor de seus atributos em um determinado momento.
- Objetos possuem IDENTIDADE, ou seja, cada objeto é diferente do outro e cada um tem seu próprio tempo de vida
- Cada objeto tem uma identidade única, mesmo que o estado seja idêntico para ambos os objetos

OBJETO

- Dados ligados ao objeto – Exemplos:
  - Círculo – ponto_centro, raio
  - linha – ponto_inicio; ponto_final
  - Cliente – Nome;Data Nascimento; Telefone
  - Telefone – Numero; Modelo; Cor
  - Exemplos:

```
//criação de objetos (sintaxe C++)
Ponto p(3,4);
Circle c(p,5.4);
Cliente pessoa;
```


MÉTODOS

- Métodos são procedimentos que determinam como o objeto se comporta.
- Através dos métodos é possível manipular os dados contidos no objeto.
- Os métodos estão ligados ao comportamento do objeto
- Exemplos
  - Um círculo poderia possuir os métodos:

draw; move; getArea; getPerimeter; setCenter

  - Um cliente poderia possuir os métodos:

calculaIdade; getTelefone

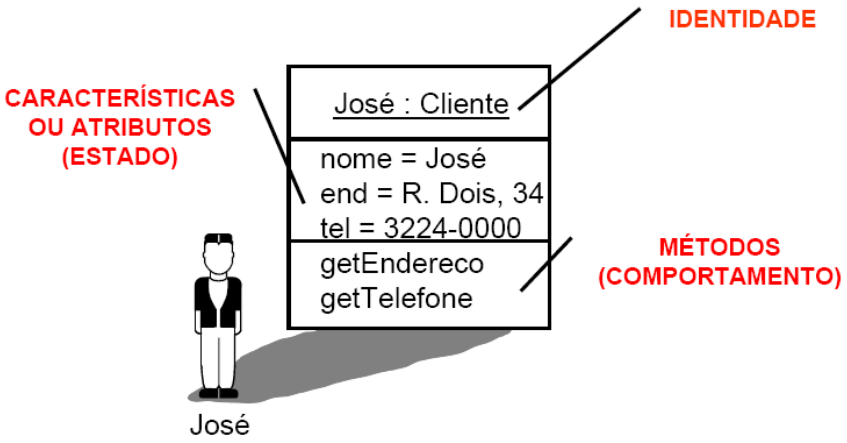
  - Um Telefone poderia possui os métodos:

tocar; discar

MÉTODOS

- Um método é a implementação de uma operação
- Métodos só tem acesso aos dados da classe para a qual foram definidos
- Métodos normalmente possuem argumentos, variáveis locais, valor de retorno, etc.
- Alguns métodos especiais:
  - Construtores – Criam objetos
  - Destrutores – Destroem objetos
  - Acessores – Recuperam o estado de um atributo (getNomeAtributo)
  - Modificadores – Alteram o estado de um atributo (setNomeAtributo)

OBJETOS - RESUMO



MENSAGEM

- Objetos se comunicam entre si através de mensagens.
- Uma mensagem é uma chamada de um método.
- A mensagem possui os seguintes componentes:
  - Receptor – nome do objeto que irá receber a mensagem
  - Método – Método do receptor que será utilizado
  - Argumentos – Informação adicional para a execução do método
  - **Exemplos**

```
Point p(0,0), pNewCenter(2,3);
Circle c(p,3);
c.getArea(); //Exemplo Mensagem
c.setCenter(pNewCenter); //Exemplo Mensagem
```


CLASSE

- Classe é um agrupamento de objetos
- A classe consiste nos métodos e nos dados que um determinado objeto irá possuir.
- Objetos são criados quando uma mensagem solicitando a criação é recebida pela sua classe.
- A programação orientada a objetos consiste em implementar as classes e na utilização das mesmas, através da sua intercomunicação.
- Um objeto é uma instância da classe.
- Os objetos de uma classe compartilham os mesmos atributos, operações, relacionamentos e semânticas
- Objetos só reagem a mensagens que fazem parte das ações do protocolo de sua classe

CLASSIFICAÇÃO

- Na POO classificação consiste em criar classes a partir dos objetos envolvidos em um determinado problema
- As classes podem ser criadas a partir do momento em que for possível isolar no domínio do problema objeto que possui atributos e métodos comuns
- Ex: Diferentes tipos de pessoas interagem com uma empresa

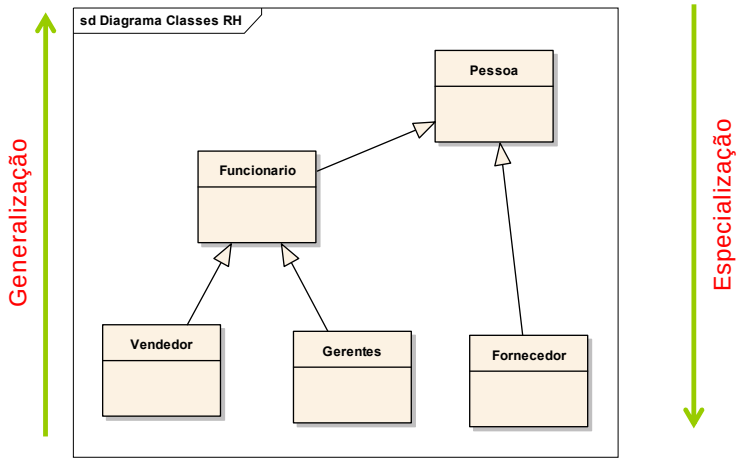
COMPORTAMENTO	CLASSE
Pessoas interessadas nos produtos	???
Pessoas que já compraram os produtos	???
Pessoas que são responsáveis por um grupo de trabalhadores	???
Pessoas responsáveis pela demonstração de produtos e sua venda	???
Trabalhadores da linha de produção	???

Generalização e Especialização

- GENERALIZAÇÃO
  - A generalização consiste em obter similaridades entre as várias classes e partir destas similaridades, novas classes são definidas.
  - Estas classes são chamadas **superclasses**
- ESPECIALIZAÇÃO
  - A especialização por sua vez consiste em observar diferenças entre os objetos de uma mesma classe e dessa forma novas classes são criadas.
  - Estas classes são chamadas **subclasses.**

Generalização e Especialização Exemplo

- Hierarquia de classes



HERANÇA

- Herança é a capacidade de uma subclasse de ter acesso as propriedades da superclasse a ela relacionada.
- Dessa forma as propriedades de uma classe são propagadas de cima para baixo em um diagrama de classes.
- Neste caso dizemos que a subclasse herda as propriedades e métodos da superclasse
- A relação de herança entre duas classes é uma relação da seguinte forma: A “e um tipo de” B, onde A e B são classes.
- Caso esta relação entre as classes não puder ser construída, em geral, também não se tem uma relação de herança entre a classe A a partir da classe B.

HERANÇA

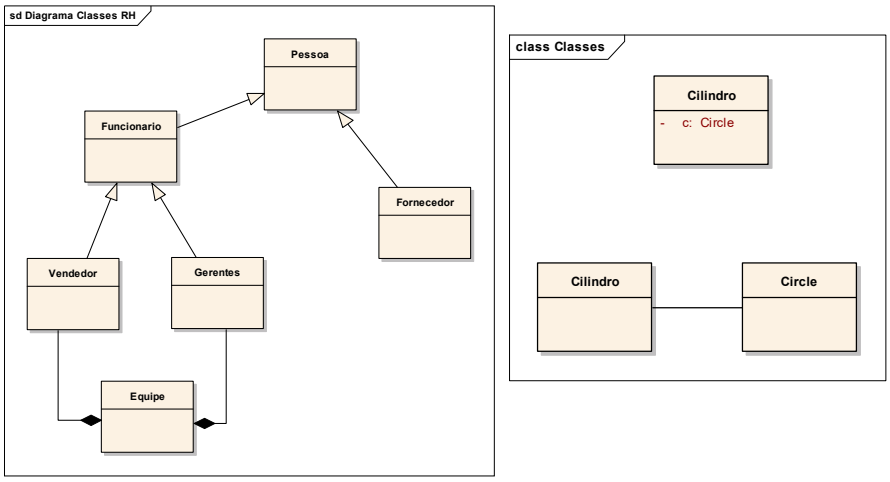
- Exemplos:
  - Um Carro de Passeio “é um tipo de ” veículo; Um caminhão “é um tipo de” veículo;
  - Um círculo “é um tipo de” uma figura geométrica; Um quadrado “é um tipo de” figura geométrica;
  - Um vendedor “é um tipo de” Empregado; Um empregado “é um tipo de” pessoa.
- Herança Múltipla
  - Uma subclasse herda características de mais uma classe
  - Exemplos
    - Um gerente de vendas “é um tipo” de vendedor e também “é um tipo de” gerente;

HERANÇA x USO

- Além da relação de herança entre as classes existe a relação de uso
- HERANÇA
  - classe A “é um tipo de” B
- USO / AGREGAÇÃO (Relação de Conteúdo)
  - classe D “contém” classe C”
  - classe D “usa” classe C”
  - classe C “é parte da” classe D
  - Exemplo: Uma **equipe contém um gerente e** um grupo de **vendedores**

Herança x Uso

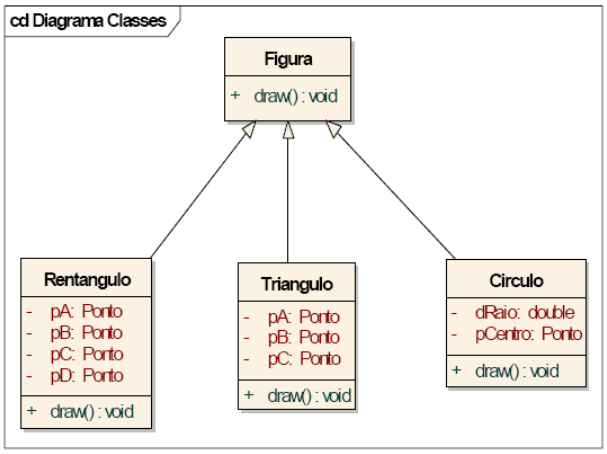
- Exemplo



POLIMORFISMO (Override)

- Os objetos respondem às mensagens que eles recebem através dos métodos.
- A mesma mensagem pode resultar em diferentes resultados. Esta propriedade é chamada de **polimorfismo**
  - **Exemplo: Método getSalario()**
    - Para um empregado qualquer → getsalario() = Salario;
    - Para o gerente → getsalario() = salario + bonificacao;
  - **Exemplo: Método draw()**
    - Para uma figura qualquer desenha uma forma não definida
    - Para o retângulo, triângulo e círculo o mesmo método responde de uma forma diferente

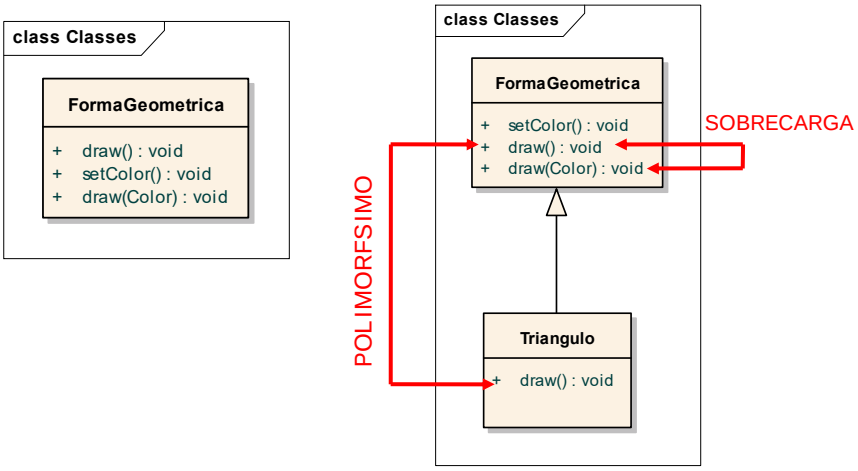
POLIMORFISMO - Exemplo



SOBRECARGA(Overload)

- Nas linguagens orientadas a objetos é possível, em uma classe, a existência de métodos que possuem o mesmo nome, porém com diferentes assinaturas.
- Este conceito é chamado de Sobrecarga (Overload)
  - **Exemplo: Em uma classe Figura, temos os seguintes métodos**
    - draw() → desenha o objeto e utiliza uma cor padrão
    - draw(Color) → desenha o objeto, porém recebe uma cor como parâmetro

SobreCarga - Exemplo



ENCAPSULAMENTO

- Conceito que indica que os dados contidos em um objeto somente poderão ser acessados e/ou modificados através de seus métodos.
- Dessa forma não é possível alterar os dados diretamente, somente através de métodos definidos no objeto
- Exemplo
  - O raio somente pode ser alterado/recuperado pelos métodos setCenter/getCenter.

ENCAPSULAMENTO

- O encapsulamento assegura que toda a comunicação com o objeto seja realizada por um conjunto pré-definido de operações
- O encapsulamento facilita as mudanças, visto que os objetos são isolados uns dos outros, reduzindo desta forma o acoplamento
- Além disso o encapsulamento facilita a manutenção de classes, bem como, garante a integridade dos atributos de um objeto em um determinado instante.

ABSTRAÇÃO

- Abstração é o processo de identificar as qualidades ou propriedades importantes do problema que está sendo modelado.
- Através de um modelo abstrato,pode-se concentrar nas características relevantes e ignorar as irrelevantes.
- Abstração é fruto do raciocínio.
- Através da abstração é possível controlar a complexidade. Isto é feito através da ênfase em características essenciais, fazendo-se uma supressão daquilo que não está ligado ao domínio do problema.

MODULARIZAÇÃO

- Consiste em decompor o problema em partes menores.
- Dessa forma o foco é mantido em itens(classes; pacotes; etc.) menores, coesos e fracamente acoplados.

Praticando os conceitos... Atividade

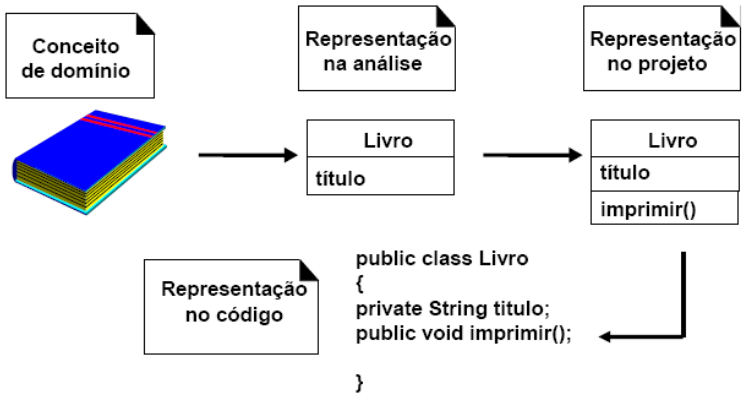
- Utilizando os conceitos
  - Herança
  - Polimorfismo
  - Sobrecarga
  - Classe
  - Atributo
  - Método
- Propor uma modelagem relacionada a um domínio de sua escolha. Por exemplo: vendas, saúde, transporte, comercio eletrônico, etc.

Análise Orientada a Objetos (OOA)
Projeto Orientado a Objetos (OOD)

- As técnicas tradicionais (Análise e Projeto Estruturado) não são adequadas para o desenvolvimento de software utilizando a orientação à objetos.
- A utilização do orientação à Objetos solicita uma nova forma de abstrair e entender o problema.
- A linguagem UML é um padrão de diagramação para visualizar os resultados da análise e projeto orientados à objetos

Análise Orientada a Objetos (OOA)
Projeto Orientado a Objetos (OOD)

- Exemplo
  - O conceito “Livro” em um sistema de biblioteca



Análise Orientada a Objetos(OOA)

- Objetivo básico
  - Identificar classes a partir das quais objetos serão representados como instâncias
- Envolve as seguintes tarefas
  - Identificação de Objetos
  - Especificação de Atributos
  - Definição de métodos
  - Comunicações entre objetos

Análise Orientada a Objetos(OOA)

- IDENTIFICAÇÃO DE OBJETOS
  - Entidades externas (Outros sistemas; dispositivos;Pessoas)
  - Coisas ligadas ao domínio do problema (Relatórios;Displays;…)
  - Ocorrências ou Eventos (Conclusão de um movimento;Alarme disparado; Clique do mouse; etc.)
  - Papéis ou funções (Engenheiro; Gerente; Vendedor) desempenhados por pessoas
  - Unidades organizacionais (Grupo; Equipe;…)
  - Lugares (Piso de fábrica; área de descarga)
  - Estruturas (Sensores; veículos de quatro rodas;…)

Análise Orientada a Objetos(OOA)

- IDENTIFICAÇÃO DE OBJETOS – CRITÉRIOS
  1. RETENÇÃO DE INFORMAÇÃO – Objeto deve guardar informação que será utilizada pelo sistema
  2. SERVIÇOS NECESSÁRIOS – Conjunto de operações identificáveis que podem mudar o valor de seus atributos
  3. MÚLTIPLOS ATRIBUTOS – Objeto deve conter mais de um atributo
  4. ATRIBUTOS COMUNS – Conjunto de atributos deve ser aplicado a todos os objetos
  5. OPERAÇÕES COMUNS – Conjunto de operações devem ser aplicáveis a todos os objetos.
  6. REQUISITOS ESSENCIAIS – Entidades externas que aparecem no espaço problema que consomem e/ou produzem informação

Análise Orientada a Objetos(OOA)

- Em uma especificação:
  - NOMES são potenciais objetos (e classes)
  - VERBOS são potenciais métodos
- A regra acima deve ser utilizada apenas como referência.
- O entendimento do contexto, das necessidades do usuário são fundamentais para classificar possíveis objetos e métodos

Análise Orientada a Objetos(OOA) Exemplo Especificação

O software SafeHome possibilita que o dono da casa configure o sistema de segurança quando ele for instalado, monitora todos os sensores ligados ao sistema de segurança e interage com o dono da casa através de um teclado (key pad) e teclas de função contidas no painel de controle do SafeHome.

Durante a instalação o painel de controle é usado para "programar" e configurar o sistema. A cada sensor é atribuído um número e um tipo, uma senha mestra é programada para armar e desarmar o sistema e números telefônicos são introduzidos para serem discados quando ocorrer um evento sensor.

Quando um evento sensor é sentido pelo software, ele dispara um alarme sonoro ligado ao sistema. Após um tempo de espera, que é especificado pelo dono da casa durante as atividades de configuração do sistema, o software disca um número telefônico do serviço de monitoração, oferece informações sobre o local, registrando a natureza do evento que foi detectado. O número será novamente discado a 20 segundos até que a ligação telefônica seja completada.

Todas as interações com o SafeHome são gerenciadas por um subsistema de interação com o usuário, que lê a entrada fornecida através do teclado e das chaves de função, exibe mensagens de prompting e informações sobre o status do sistema no mostrador de cristal líquido (LCD). A interação com o teclado assume a seguinte forma...

Análise Orientada a Objetos(OOA) Exemplo Especificação

O software SafeHome possibilita que o **dono da casa** configure o **sistema de segurança** quando ele for instalado, monitora todos os **sensores** ligados ao **sistema de segurança** e interage com o **dono da casa** através de um **teclado** (key pad) e **teclas de função** contidas no **painel de controle** do SafeHome.

Durante a instalação o **painel de controle** é usado para "programar" e configurar o **sistema**. A cada **sensor** é atribuído um **número** e um **tipo**, uma **senha mestra** é programada para armar e desarmar o **sistema** e **números telefônicos** são introduzidos para serem discados quando ocorrer um **evento sensor**.

Quando um **evento sensor** é sentido pelo software, ele dispara um **alarme sonoro** ligado ao **sistema**. Após um **tempo de espera**, que é especificado pelo **dono da casa** durante as atividades de configuração do sistema, o software disca um **número telefônico** do **serviço de monitoração**, oferece **informações sobre o local**, registrando a **natureza do evento** que foi detectado. O número será novamente discado a 20 segundos até que a ligação telefônica seja completada.

Todas as interações com o SafeHome são gerenciadas por um **subsistema** de interação com o **usuário**, que lê a **entrada** fornecida através do **teclado** e das **chaves de função**, exibe **mensagens** de prompting e informações sobre o status do **sistema** no **mostrador de cristal líquido** (LCD). A interação com o teclado assume a seguinte forma...

Análise Orientada a Objetos(OOA)

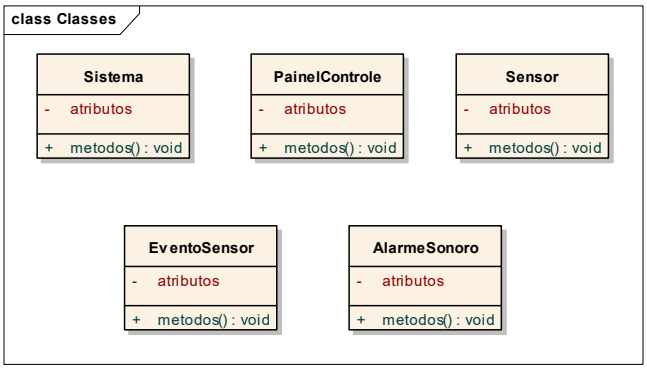
NOME	CRITÉRIO
dono da casa	Papel ou entidade externa
sistema de segurança	Coisa
sensores	Entidade externa
teclado	Entidade externa
teclas de função	Entidade externa
painel de controle	Entidade externa
número	Atributo do sensor
Tipo	Atributo do sensor
senha mestra	Coisa
números telefônicos	Coisa
evento sensor	Ocorrência
alarme sonoro	Entidade externa
tempo de espera	Atributo do sistema
Serviço de monitoração	Unidade Organizacional ou Ent. Externa
informações sobre o local	Atributo do sistema
natureza do evento	Atributo do sistema
subsistema	Entidade externa
entrada	Entidade externa
chaves de função	Entidade externa
mensagens	Entidade externa
mostrador de cristal líquido	Entidade externa

Análise Orientada a Objetos(OOA)

NOME	CRITÉRIO
dono da casa	Rejeitado: 1 e 2 falham embora 6 se aplique
sistema de segurança	Aceito: Todas se aplicam
sensores	Aceito: Todas se aplicam
teclado	Aceito: Todas se aplicam
teclas de função	Aceito: Todas se aplicam
painel de controle	Aceito: Todas se aplicam
número	Rejeitado: 3 falha
Tipo	Rejeitado: 3 falha
senha mestra	Rejeitado: 3 falha
números telefônicos	Rejeitado: 3 falha
evento sensor	Aceito: Todas se aplicam
alarme sonoro	Aceito: 2, 3, 4, 5 e 6
tempo de espera	Rejeitado: 3 falha
Serviço de monitoração	Rejeitado: 1 e 2 falham embora 6 se aplique
informações sobre o local	Rejeitado: 3 falha
natureza do evento	Rejeitado: 3 falha
subsistema	Aceito: Todas se aplicam
entrada	Rejeitado: 3 falha
chaves de função	Aceito: Todas se aplicam
mensagens	Aceito: Todas se aplicam
mostrador de cristal líquido	Aceito: Todas se aplicam

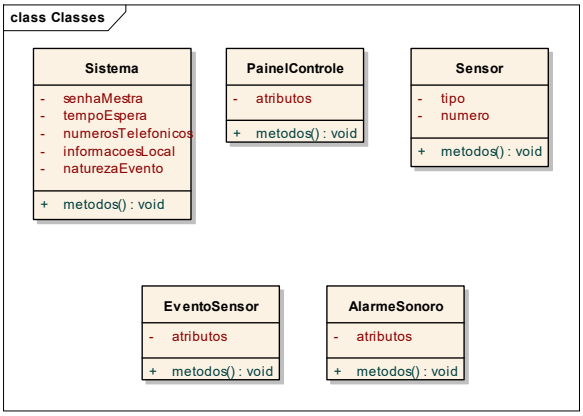
Identificação de Objetos

- Os objetos do painel de controle serão considerados separadamente.
- Os objetos abaixo são o ponto de partida para o desenvolvimento do sistema
- Objetos ainda não possuem atributos e métodos



Especificação de Atributos

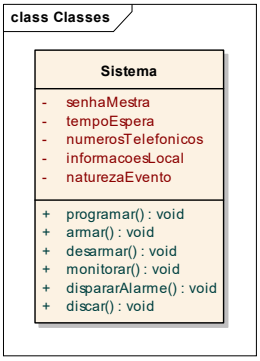
- Necessário estudar e analisar a descrição do sistema
- Pergunta Importante – "Quais informações definem este objeto?"



Definição dos Métodos

- Necessário estudar e analisar a descrição do sistema
- Verbos são potenciais operações

Configure	sentido
Instalado	dispara
Monitora	especificado
Ligados	configuração
Interage	disca
Instalação	registrando
"programar"	detectado
configurar	Discado
programada	ligação
armar	interações
desarmar	gerenciadas
introduzidos	interação
serem discados	lê
ocorrer	fornecida
	exibe



Análise Orientada a Objetos(OOA)

O departamento de obras públicas da cidade de Uberlândia decidiu desenvolver um sistema de computador para rastreamento e conserto de buracos de rua (SIRCOB).

À medida que são registrados buracos de rua, eles recebem um número de identificação e são armazenados de acordo com o endereço da rua, tamanho (numa escala de 0 a 10), localização (no meio dar rua; na calçada; etc.), bairro (determinado a partir do endereço da rua) e prioridade de reparo (determinada a partir do tamanho do buraco).

Dados de ordem de trabalho são associados a cada buraco, e eles incluem localização e tamanho do buraco, número de identificação da equipe de reparos, número de pessoas na equipe, equipamentos designados, horas aplicadas ao reparo, status do trabalho (em andamento, concluído, não concluído), quantidade de material de enchimento usado e custo do reparo (computado a partir das horas trabalhadas, número pessoas, material e equipamentos usados).

Finalmente, um arquivo de danos ocorridos é criado para guardar informações sobre danos registrados devido ao buraco, o qual inclui o nome do cidadão, endereço, número telefônico, tipo de dano e a quantia em reais a ser paga. O SIRCOB é um sistema on-line; as consultas devem ser feitas interativamente.

Análise Orientada a Objetos(OOA)

O departamento de obras públicas da cidade de Uberlândia decidiu desenvolver um **sistema** de computador para rastreamento e conserto de **buracos** de rua (SIRCOB).

À medida que são registrados buracos de rua, eles recebem um **número de identificação** e são armazenados de acordo com **o endereço da rua**, **tamanho** (numa escala de 0 a 10), **localização** (no meio dar rua; na calçada; etc.), **bairro** (determinado a partir do endereço da rua) e **prioridade** de reparo (determinada a partir do tamanho do buraco).

Dados de **ordem de trabalho** são associados a cada buraco, e eles incluem **localização** e **tamanho do buraco**, **número de identificação da equipe de reparos**, **número de pessoas na equipe**, **equipamentos** designados, **horas** aplicadas ao reparo, **status** do trabalho (em andamento, concluído, não concluído), **quantidade** de material de enchimento usado e custo do reparo (computado a partir das horas trabalhadas, número pessoas, material e equipamentos usados).

Finalmente, um **arquivo** de **danos** ocorridos é criado para guardar informações sobre danos registrados devido ao buraco, o qual inclui o **nome do cidadão**, **endereço**, **número telefônico**, **tipo de dano** e a **quantia** em reais a ser paga. O SIRCOB é um sistema on-line; as consultas devem ser feitas interativamente.

Identificação dos Objetos

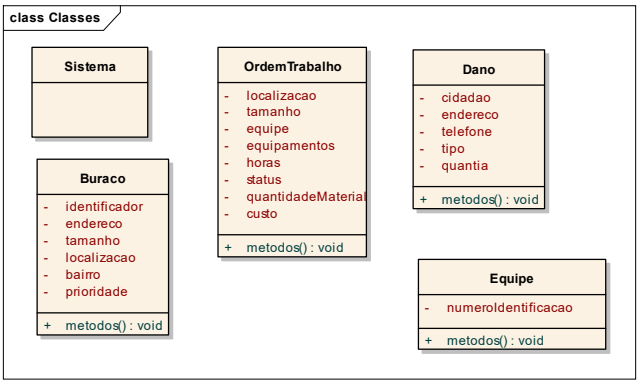
NOMES	CLASSIFICAÇÃO	ANÁLISE
departamento	Unidade Organizacional	Rejeitado: 1 e 2 Falham
sistema	**coisa**	**Aceito: Todas se aplicam**
buracos	**coisa**	**Aceito: Todas se aplicam**
número de identificação	coisa	Rejeitado: 3 falha
endereço da rua	coisa	Rejeitado: 3 falha
tamanho	coisa	Rejeitado: 3 falha
localização	coisa	Rejeitado: 3 falha
bairro	bairro	Rejeitado: 3 falha
prioridade	coisa	Rejeitado: 3 falha
ordem de trabalho	**coisa**	**Aceito: Todas se Aplicam**
número de identificação da equipe	coisa	Rejeitado: 3 falha
número de pessoas	coisa	Rejeitado: 3 falha
equipamentos	coisa	Rejeitado: 3 falha
horas	coisa	Rejeitado: 3 falha
quantidade	coisa	Rejeitado: 3 falha
custo	coisa	Rejeitado: 3 falha
arquivo de danos	**estrutura**	**Aceito: Todas se aplicam**
nome do cidadão	coisa	Rejeitado: 3 falha
Endereço	coisa	Rejeitado: 3 falha
número telefônico	coisa	Rejeitado: 3 falha
tipo de dano	coisa	Rejeitado: 3 falha

Identificação dos Atributos

NOMES	CLASSIFICAÇÃO	ANÁLISE
departamento	Unidade Organizacional	Rejeitado: 1 e 2 Falham
sistema	**coisa**	**Aceito: Todas se aplicam**
buracos	**coisa**	**Aceito: Todas se aplicam**
número de identificação	coisa	Rejeitado: 3 falha (ATRIBUTO)
endereço da rua	coisa	Rejeitado: 3 falha (ATRIBUTO)
tamanho	coisa	Rejeitado: 3 falha (ATRIBUTO)
localização	coisa	Rejeitado: 3 falha (ATRIBUTO)
bairro	bairro	Rejeitado: 3 falha (ATRIBUTO)
prioridade	coisa	Rejeitado: 3 falha (ATRIBUTO)
ordem de trabalho	**coisa**	**Aceito: Todas se Aplicam**
número de identificação da equipe	coisa	Rejeitado: 3 falha (ATRIBUTO)
número de pessoas	coisa	Rejeitado: 3 falha (ATRIBUTO)
equipamentos	coisa	Rejeitado: 3 falha (ATRIBUTO)
horas	coisa	Rejeitado: 3 falha (ATRIBUTO)
quantidade	coisa	Rejeitado: 3 falha (ATRIBUTO)
custo	coisa	Rejeitado: 3 falha (ATRIBUTO)
arquivo de danos	**estrutura**	**Aceito: Todas se aplicam**
nome do cidadão	coisa	Rejeitado: 3 falha (ATRIBUTO)
Endereço	coisa	Rejeitado: 3 falha (ATRIBUTO)
número telefônico	coisa	Rejeitado: 3 falha (ATRIBUTO)
tipo de dano	coisa	Rejeitado: 3 falha (ATRIBUTO)

Identificando Classes e Atributos

- Além dos objetos mostrados a seguir que são identificados diretamente a partir da especificação do sistema outros objetos podem ser necessários dependendo de como o sistema será construído. (Ex.: Equipe; Trabalhador; etc.)



Identificando os Métodos

- Verbos destacados
  - registrados
  - Recebem
  - Armazenados
  - Associados
  - Guardar
  - Consultar
- Outras operações serão necessárias para que o sistema funcione corretamente